HiperSense: An Integrated System for Dense Wireless Sensing and Massively Scalable Data Visualization

Pai H. Chou[1,2,4], Chong-Jing Chen[1,2], Stephen F. Jenks[2,3], and Sung-Jin Kim[3]

[1] Center for Embedded Computer Systems,University of California, Irvine, CA
[2] Electrical Engineering and Computer Science, University of California Irvine, CA
[3] California Institute for Telecommunications and Information Technology, Irvine, CA
[4] Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan

Abstract. HiperSense is a system for sensing and data visualization. Its sensing part is comprised of a heterogeneous wireless sensor network (WSN) as enabled by infrastructure support for handoff and bridging. Handoff support enables simple, densely deployed, low-complexity, ultra-compact wireless sensor nodes operating at non-trivial data rates to achieve mobility by connecting to different gateways automatically. Bridging between multiple WSN standards is achieved by creating virtual identities on the gateways. The gateways collect data over Fast Ethernet for data post-processing and visualization. Data visualization is done on HIPerWall, a 200-megapixel display wall consisting of 5 rows by 10 columns of 30-inch displays. Such a powerful system is designed to minimize complexity on the sensor nodes while retaining high flexibility and high scalability.

1 Introduction

Treating the physical world as part of the cyber infrastructure is no longer just a desirable feature. *Cyber-physical systems* (CPS) are now the mandate of many national funding agencies worldwide. CPS entails more than merely *interfacing* with the physical world. The goal is to form synergy between the cyber and the physical worlds by enabling cross pollination of many more features.

A wireless sensor network (WSN) is an example of cyber-physical interface. A sensor converts a physical signal into a quantity to enable further processing and interpretation by a computing machine. However, it is still mostly an *interface*, rather than a *system* in the CPS sense. Most WSNs today lack the cyber part, which would leverage the vast amount of information available on the network to synthesize new views of data in ways never possible before. An example of a system that is one step towards CPS is SensorMap [1], which offers a GoogleEarth-style application to be augmented with sensor data collected at the corresponding positions. SensorMap provides an *abstraction* in the form of the sensor-to-map programming interface (API), so that data providers can leverage the powerful cloud-computing backend without having to re-invent yet another tool for each highly specialized application.

However, data visualization can be much more than merely superimposing data on geographical or topological maps by a cloud computing system to be rendered on a personal computer. In fact, the emergence of large, high-resolution displays, high-performance workstations, and high-speed interconnection interfaces give rise to large *display walls* as the state-of-the-art visualization systems. An example is the HIPerWall, a 200-megapixel tiled screen driven by 25 PowerMac G5s interconnected by two high-speed networks [2, 3]. Such a system has found use in ultra-high-resolution medical imaging and appears as a prime candidate for visualization of a wide variety of sensor data as well.

This paper describes work in progress on such a massive-scale sensing and visualization system, called HiperSense. On the sensing side, we aim to further develop a scalable infrastructure support, called EcoPlex [4]. It consists of a tiered network, where the upper tier contains the gateways and the lower tier includes the sensor nodes. The gateways support handoff for mobility and bridging of identity for integrating heterogeneous radio and networking standards. On the data visualization side, we feed the data to the HIPerWall, which can then render the data in an interactive form across 50 screens as one logical screen. This paper reports on the technologies developed to date and discusses practical issues that we have encountered.

2 Related Work

Several multiple-access protocols that use multiple frequency channels have been proposed for wireless sensor networks [5, 6]. Some have been evaluated only by simulation, while others have been adopted for researchers in ad-hoc network domains. Many protocols for WSNs have been implemented on the popular MicaZ or TelosB platforms.

Y-MAC [7] is a TDMA-based protocol that schedules the receivers in the neighborhood by assigning available receiving time slots. Its light-weight channel-hopping mechanism can enable multiple pairs of nodes to communicate simultaneously, and it has been implemented on the RETOS operating system running on a TmoteSky-class sensor node. However, a problem with Y-MAC is the relatively low performance due to time synchronization and overhead of channel-switching time. Base on previous experimental results from another team [8] on the Chipcon CC2420 radio transceiver, which implements the IEEE 802.15.4 MAC as used on MicaZ and TelosB, the channel switching time is nearly equal to the time it takes to transmit a 32-byte data packet. Therefore, changing to another frequency channel dynamically and frequently can become a damper on system performance.

Le et al proposed and implemented another multi-channel protocol on the MicaZ [8]. Their protocol design does not require time synchronization among the sensor nodes. They also take the channel-switching time into consideration for sensor nodes that are equipped with only a half-duplex radio transceiver. A distributed control algorithm is designed to assign accessible frequency channels for each sensor node dynamically to minimize the frequency-changing time. The compiled code is around 9.5 KB in ROM and 0.7 KB in RAM on the top of TinyOS. Although it is smaller than others' solutions, it is still too big to fit in either the Eco or μPart sensor node, both of which have much smaller RAM and ROM.

After data collection, showing sensing data in a meaningful way in real-time is another emerging issue. Several solutions have been proposed to integrate the sensing data with a geographic map [9, 10]. SensorMap [1] from Microsoft Research is for displaying sensor data from SenseWeb [11] on a map interface. They are able to provide tools to query sensor nodes and visualize sensing data in real-time. Google Map with traffic can show not only the live traffic but also the history traffic at day and time [12]. However, these works currently assumes limited screen resolution and has not been scaled to the 200-megapixel resolution of the HIPerWall.

3 System Architecture

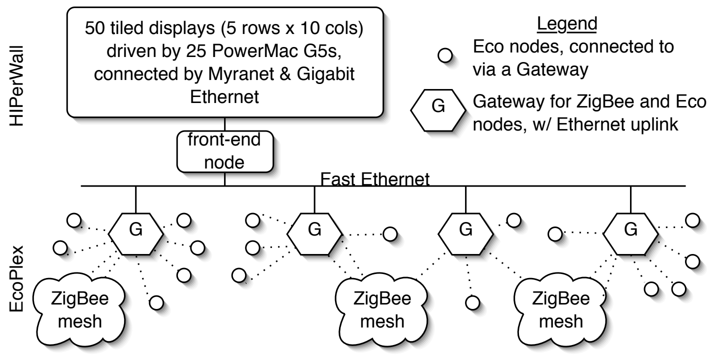

HIPerWall
50 tiled displays (5 rows x 10 cols) driven by 25 PowerMac G5s, connected by Myranet & Gigabit Ethernet
front-end node
Fast Ethernet
Legend
Eco nodes, connected to via a Gateway
G
Gateway for ZigBee and Eco nodes, w/ Ethernet uplink
EcoPlex
ZigBee mesh

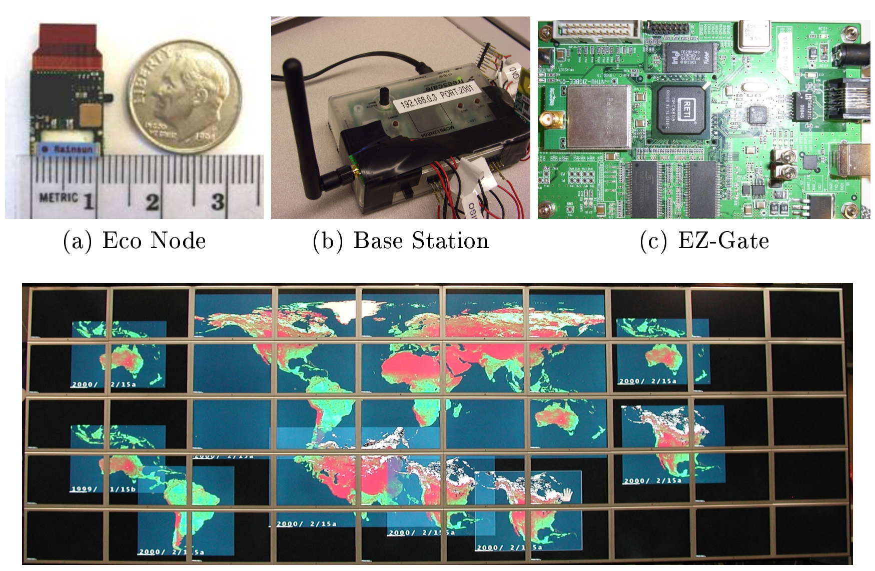
Fig. 1. The HiperSense architecture.

Fig. 1 shows the architecture of HiperSense. It consists of HIPerWall as the visualization subsystem and EcoPlex as the sensing infrastructure. This section summarizes each subsystem and their interconnection.

3.1 HIPerWall

HIPerWall is a tiled display system for interactive visualization, as shown in Fig. 2(d). The version we use consists of fifty 30-inch LCD monitors arranged in five rows by ten columns. Each monitor has a pixel count of 2560 $\times$ 1600 at 100 dots per inch of resolution, and therefore the entire HIPerWall has a total resolution of 204.8 million pixels. The tiled display system is driven by 25 PowerMac G5 workstations interconnected by two networks to form a high-performance computing cluster. One network uses Myrinet for very high-speed data transfer, and the other network uses Gigabit Ethernet for control. The HIPerWall software is portable to other processors and operating systems, and it can be configured for a wide variety of screen dimensions.

A user controls the entire HIPerWall from a separate computer called the *front-end node*. It contains a reduced view of the entire display wall, enabling the user to manipulate the display across several screens at a time. The front-end node also serves as the interface between the sensing subsystem and the visualization subsystem.



(a) Eco Node (b) Base Station (c) EZ-Gate

(d) HIPerWall

Fig. 2. Components of HiperSense

3.2 EcoPlex

EcoPlex is a tiered, infrastructure-based heterogeneous wireless sensor network system. Details of EcoPlex can be found in another paper [4], and here we highlight the distinguishing features. At the bottom tier are the wireless sensor nodes. The top tier consists of a network of gateway nodes.

Lower Tier Sensor Nodes EcoPlex currently supports two types of nodes with different communication protocols: ZigBee and Eco. Our platform allows other protocols such as Bluetooth and Z-Wave to be bridged without any inherent difficulty.

ZigBee is a wireless networking protocol standard primarily targeting low-duty-cycle wireless sensing applications [13]. In recent years, it is also targeting home automation domain. ZigBee is a network protocol that supports ad hoc mesh networking, although it also defines roles for not only end devices but also routers and coordinators. ZigBee is built on top of the IEEE 802.15.4 media access control (MAC) layer, which is based on carrier-sense multiple access with collision avoidance (CSMA/CA). Currently, many wireless sensor applications are built on top of 802.15.4, though not necessarily with the ZigBee protocol stack, since it occupies about 64–96KB of program memory.

Another type of wireless sensor node supported in EcoPlex is Eco [14, 15], our own ultra-compact wireless sensing platform, as shown in Fig. 2(a). It is 1 cm^3 in volume including the MCU, RF, antenna, and sensor devices. It contains 4 KB RAM and 4 KB EEPROM. The radio is based on Nordic VLSI's ShockBurst protocol at 1 Mbps, and it is a predecessor of Wibree, also known as Bluetooth Low Energy Technology as a subset of Bluetooth 3.0 [16]. Eco is possibly the world's smallest, self-contained, programmable, expandable platform to date. A flex-PCB connector enables an Eco node to be connected to other I/O devices and power. Eco is meant to complement ZigBee in that Eco nodes can be made much smaller and cheaper than ZigBee ones, and thus they can be deployed where ZigBee nodes cannot, especially in some highly wearable or size-constrained applications.

Upper Tier: Gateways The upper tier of EcoPlex consists of a network of gateway nodes called EZ-Gates. An EZ-Gate is essentially a Fast Ethernet router based on an ARM-9-core network processor running Linux 2.6. It is augmented with the radio transceivers that are needed for supporting the protocols used by the wireless sensor nodes. In this case, one ZigBee transceiver and two Eco transceivers are added to each EZ-Gate. For ZigBee support, the EZ-Gate implements the protocol stack of a ZigBee coordinator. Since Eco is more resource-constrained and can afford to implement only a much simpler protocol, the gateway provides relatively more support in the form of *handoff* and *virtual identity*.

Eco nodes connect to the gateways and not to each other, the same way cellular phones connect to base stations but not to each other. Just as cellular towers perform handovers based on the proximity of the mobile, our EZ-Gates perform handoffs based on the link quality of the Eco nodes. Unlike cell phones, which are treated as independently operated units, EcoPlex supports the concepts of clusters, which are groups of wireless sensor nodes that work together, are physically close to each other, and move as a group [17]. Instead of performing handoff for each node individually, cluster handoff would rely on a node as a representative for the entire cluster and has been shown to be effective especially in dense deployments. EZ-Gates also support bridging in the form of virtual identity. That is, for every Eco node connected to EcoPlex, the owner gateway maintains a node identity in the ZigBee space. This way, an Eco node appears just like any other ZigBee node and can communicate logically with other ZigBee nodes, without having to be burdened with the heavy ZigBee stack.

A simpler base station without the handoff and virtual identity support was also developed. It is based on the Freescale DEMO9S12NE64 evaluation board connected to a Nordic nRF24L01 transceiver module, as shown in Fig. 2(b). It has a Fast Ethernet uplink to the front-end node. It was used for the purpose of developing code between the Ethernet and Eco sides before porting to the EZ-Gate for final deployment.

4 System Integration

HiperSense is more than merely connecting the EcoPlex sensing subsystem with the HIPerWall tiled display system. It entails design of a communication and execution scheme to support the needs of sensing and visualization. This section first discusses considerations for HiperSense to support CPS-style visualization. Then, we describe the communication scheme for system integration.

4.1 Visualization Styles and Support

Visualization is the graphical rendering of data in a form that helps the user gain new insights into the system from which the data is collected. Unlike many other visualization systems that render only static data that has been collected and stored in advance, HiperSense is designed to support visualization of both static data files and live data streams. More importantly, we envision a visualization system that synthesizes views from static or live data and other sources available on the Internet.

As an example, consider a WSN that collects vibration data from sensor nodes on a pipeline network. The user is not interested in vibration per se but wants to non-invasively measure the propagation speed of objects traveling inside the pipeline based on the peak vibration. In this case, time-history plots of raw vibration data is not so meaningful to the user; instead, the data streams must be compared and processed to derive the velocity. The visualization system then renders the velocity by superimposing colored-encoded regions over high-resolution images of the pipeline network and its surroundings. Moreover, the user may want the ability to not only navigate spatially similar to GoogleEarth but also temporally by seeing how the peak velocity shifts over time.

To support smooth spatial navigation, HIPerWall relies on replication or prefetching of large data (e.g., patches of GoogleEarth photos) from adjacent nodes. The data could be information in the local database system, images or videos. These data shown on the screens are treated as independent objects and can be zoomed in, zoomed out or rotated arbitrarily. The front-end node simply sends out commands to every computing node to indicate which object should be displayed, which position the object is and the other properties to control the object. This mechanism reduces the traffic between the front-end node and the cluster of computing nodes.

To support real-time access to data, supervisory control and data acquisition (SCADA) systems, which are commonly found in factories for up to thousands of sensing and actuation channels per site, have used real-time databases for the purpose of logging and retrieval of data. A similar set up can be built for HiperSense. Historic data can also be retrieved from the real-time database via a uniform interface. However, one difference between a conventional SCADA and HiperSense is that the former is commonly handled by one or a small number of computers, while the latter relies on a cluster of 25 computers to parallelize the handling of the massive graphical bandwidth.

For the purpose of tiled visualization of live sensor data, we program the front-end node of the HIPerWall to also be a fat client for data collection from wireless sensor nodes via the gateways in EcoPlex. The front-end node then broadcasts the collected data to the cluster of computing nodes inside HIPer-Wall. Every computing node decodes the whole data packet but shows only the portion that is visible on the two LCD screens that it controls. This broadcasting mechanism removes the time synchronization between all workstations and ensures that all sensing data can be shown on the tiled displays at the same time. If the backbone of Intranet and the cluster of workstations both support Jumbo Frame [18], we can increase the overall system performance and deploy more wireless sensor nodes at once.

4.2 Protocols for the Tiers

EcoPlex currently supports both ZigBee and Eco as two complementary wireless protocols. The ZigBee standard is designed for sporadic, low-bandwidth communication in an ad hoc mesh network, whereas Eco is capable of high-bandwidth, data-regular communication on ultra-compact hardware in a star network. Of course, it is possible for each platform to implement each other's characteristics, but they would be less efficient. For the purpose of integration with HiperSense, ZigBee does not pose a real problem due to its lack of timing constraints. We therefore concentrate our discussion on integration of Eco nodes and gateways.

On many wireless sensor nodes, the software complexity is dominated by the protocol stack. The code sizes of the typical protocol stacks of Bluetooth, ZigBee, and Z-Wave are 128KB, 64–96KB, and 32KB, respectively, whereas the main program of a sensing application can be as little as a few kilobytes. Our approach to minimizing complexity on the Eco nodes is to externalize it: we implement only the protocol-handling *mechanisms* on Eco and move most *policies* out to the infrastructure. This can be accomplished by making nodes passive with a *thin-server, fat-client* organization. That is, the sensor nodes are passive and the host actively pulls data from them. This effectively takes care of arbitration[5] and effective acknowledgment. The core mechanism can be implemented in under 40 bytes of code. After adding support for multi-gateway handoff and joining,

[5] no intra-network collision unless there is a malfunctioning node

channel switching, and a number of other performance enhancement policies (e.g., amortize the pulling overhead by returning multiple packets), the code size can still be kept around 2KB.

Once the reliable communication primitives are in place, then we can add another layer of software for dynamic code update and execution [19] on these nodes. Our vision is *host-assisted dynamic compilation*, where the host or its delegate dynamically generates optimized code to be loaded into the node for execution. This will be much more energy efficient than a general-purpose protocol stack that must anticipate all possibilities. An example is the protocol stack for network routing. Since our gateway as well as many commercially available gateways run Linux and have plenty of storage available, the gateways should also be able run synthesizer, compiler, and optimizer without difficulty.

The front-end node of the HIPerWall acts as a client to query data from all gateways. Each gateway is connected to the front-end node via a wired interface with higher bandwidth than the wireless interface. At beginning, all wireless sensor nodes communicate with the gateway via the control channel. The front-end node issues frequency switching commands to the wireless sensor nodes based on the sampling rate of each wireless sensor node and the available bandwidth of each wireless frequency channel. Later on, the front-end node issues a command packet to the gateway to get data from the wireless sensor nodes controlled by the gateway. The gateway in turn broadcasts the command packet to all Eco sensor nodes on the gateway's own frequency channel. The gateway packs all pulled data together and forwards the data to the front-end node, which in turn broadcasts the data to the real-time databases for visualization. ZigBee nodes push data rather than being pulled. This way, the ZigBee network can coexist with the Eco network by sporadically taking up bandwidth only when necessary.

For HiperSense, the front-end node can resend the command packet to a wireless sensor node if it does not get any reply packet within certain amount of time. In order to improve the system performance, we have the retransmission mechanism inside the gateways instead of having it at the front-end node. A gateway resends the pulling command packet that it received from the front-end node if it does not receive any reply packet from a wireless sensor node after a pre-defined timeout period.

5 Evaluation

This section presents experimental results on a preliminary version of HiperSense. The experimental setup consists of 100 Eco nodes (Fig. 3(a)) and two gateways densely deployed in an area from 2 to 16 m^2. The larger setup is for a miniature-scale water pipe monitoring system, where nodes measure vibration at different junctions. The gateways are connected to the HIPerWall's front-end node (Fig. 3(b)) via Fast Ethernet. We compare the performance of our system with other works in terms of measured throughput, latency, and code sizes for different sets of features included.

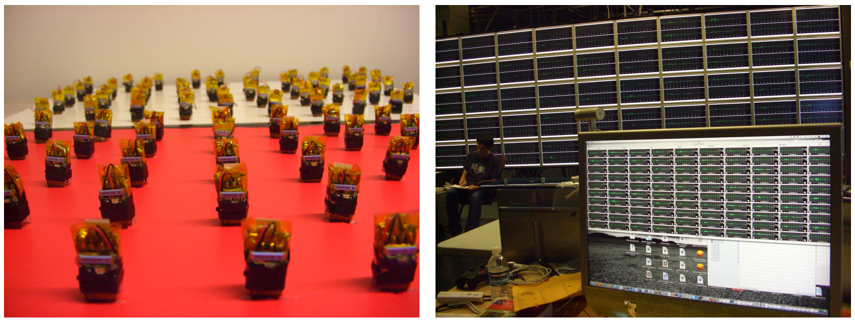
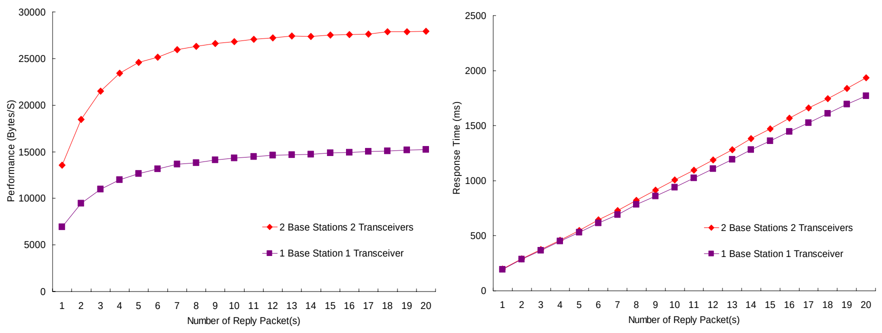

(a) 100 Eco Nodes (b) Front-End Node and HIPerWall

Fig. 3. Experimental Setup

5.1 Latency and Throughput



(a) Throughput (b) Latency

Fig. 4. Performance Results

Fig. 4 shows the measured aggregate throughput and per-node latency results for one and two gateways over different numbers of reply packets per pull. Returning more packets per pull enables amortization of pulling overhead, though at the expense of increased latency. The lower and upper curves for each chart shows the result for one and two gateways, respectively. In the case of one gateway, the throughput ranges from 6.9 KB/s to 15.8 KB/s for 20 reply packets per pull, though the latency increases linearly from around 100 ms to over 1.5 seconds. By doubling the number of gateways, the aggregate throughput ranges from 13.5KB/s for one reply per pull to 27.9 KB/s for 20 reply packets per pull. In the latest case, the total throughput increases by 81% while the latency increases by 9%.

The data rate is rather low compared to the bandwidth within HIPerWall. With nonoverlapping frequencies, EcoPlex can scale up to 50 nodes/gateway $\times$

14 channels/gateway = 6200 nodes with only 0.1% utilization of the bandwidth on the Gigabit Ethernet or 1% of Fast Ethernet.

5.2 Comparison

The closest work to ours in the area of communication protocol for the wireless sensor nodes is the Multi-Channel MAC (MCM) protocol as proposed by Le et al from the Cyber Physical Computing Group [8]. Their protocol was built on the top of TinyOS v2.x, whose minimum code size is around 11 KB [20]. Depending on the hardware and software configurations, the compiled code size of TinyOS v2.x could exceed 20 KB [20].

Table 1. Comparison Between HiperSense for Eco nodes and the Multi-Channel MAC protocol for TinyOS 2 [8].

Code Size	HiperSense	MCM Protocol
MAC layer	31 bytes	9544 bytes
Runtime Support	1.1 KB	11-20 KB
Dynamic Execution	430 bytes	N/A
Total Code Size	1.56 KB + loaded code	20.5-29.5 KB

Table 1 shows the required code size in the ROM after compilation. In HiperSense, the gateways and the front-end node handle most of the protocol policies originally handled by the sensor nodes. This enables the sensor nodes to be kept minimally simple with only the essential routines. Moreover, the processing time on a sensor node can be shortened, and the firmware footprint in the ROM is also minimized. We implemented a dynamic loading/dispatching layer, which occupies 430 bytes, enabling the node to dynamically load and execute code fragments that can be highly optimized to the node in its operating context [19]. In contrast, the MCM protocol occupies 9544 bytes on the top of TinyOS, which can increase the total code size to 29.5KB. That is over an order of magnitude larger than our code size.

6 Conclusion and Future Work

This paper reports the progress on the HiperSense sensing and visualization system. The sensing aspect is based on EcoPlex, which is an infrastructure-based, tiered network system that supports heterogeneous wireless protocols for interoperability and handoff for mobility. We keep node complexity low by implementing only bare minimum mechanisms and either externalize the policies to the host side or make them dynamically loadable. The visualization subsystem is based on HIPerWall, a tiled display system capable of rendering 200 megapixels of display data. By feeding the data streams from EcoPlex to the front-end node of the HIPerWall and replicating them among the nodes within HIPerWall, we are making possible a new kind of visualization system. Unlike previous applications that use static data, now we can visualize both live and historic data. Scalability in a dense area was shown with 100 wireless sensor nodes in a 2m^2 area. By utilizing all frequency channels, we expect HiperSense to handle 6200 independent streams of data. Applications include crowd tracking, miniature-scale pipeline monitoring, and a wide variety of medical applications. Future work includes making the protocol more adaptive and power manageable on the wireless sensor nodes. Dynamic code loading and execution has been implemented but still relies on manual coding, and it is a prime candidate for automatic code synthesis and optimization.

Acknowledgments

The authors would like to thank Seung-Mok Yoo, Jinsik Kim, and Qiang Xie for their assistance with this work on the Eco protocol, and Chung-Yi Ke, Nai-Yuan Ko, Chih-Hsiang Hsueh, and Chih-Hsuan Lee for their work on EcoPlex. The authors also would like to thank Duy Lai for his assistance on HIPerWall. This research project is sponsored in part by the National Science Foundation CAREER Grant CNS-0448668, UC Discovery Grant itl-com05-10154, the National Science Council (Taiwan) Grant NSC 96-2218-E-007-009, and Ministry of Economy (Taiwan) Grant 96-EC-17-A-04-S1-044. HIPerWall was funded through NSF Major Research Instrumentation award number 0421554. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Nath, S., Liu, J., Miller, J., Zhao, F., Santanche, A.: SensorMap: a web site for sensors world-wide. In: SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems, New York, NY, USA, ACM (2006) 373–374

[2] Kuester, F., Gaudiot, J., Hutchinson, T., Imam, B., Jenks, S., Potkin, S., Ross, S., Sorooshian, S., Tobias, D., Tromberg, B., Wessel, F., Zender, C.: HIPerWall: A high-performance visualization system for collaborative earth system sciences. `http://dust.ess.uci.edu/prp/prp_Kue04.pdf` (2004)

[3] Jenks, S.: HIPerWall. `http://hiperwall.calit2.uci.edu/`

[4] Ke, C.Y., Ko, N.Y., Hsueh, C.H., Lee, C.H., Chou, P.H.: EcoPlex: Empowering compact wireless sensor platforms via roaming and interoperability support. In: Proceedings of the Sixth Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services (MobiQuitous 2009), Toronto, Canada (July 13–16 2009)

[5] Wu, S.L., Lin, C.Y., Tseng, Y.C., Sheu, J.P.: A new multi-channel MAC protocol with on-demand channel assignment for multi-hop mobile ad hoc networks. ISPAN (2000) 232

[6] So, H.S.W., Walrand, J., Mo, J.: McMAC: A parallel rendezvous multi-channel MAC protocol. Wireless Communications and Networking Conference (March 2007) 334–339

[7] Kim, Y., Shin, H., Cha, H.: Y-MAC: An energy-efficient multi-channel MAC protocol for dense wireless sensor networks. In: IPSN '08: Proceedings of the 2008 International Conference on Information Processing in Sensor Networks, Washington, DC, USA, IEEE Computer Society (2008) 53–63

[8] Le, H.K., Henriksson, D., Abdelzaher, T.: A practical multi-channel media access control protocol for wireless sensor networks. In: IPSN '08: Proceedings of the 2008 International Conference on Information Processing in Sensor Networks (ipsn 2008), Washington, DC, USA, IEEE Computer Society (2008) 70–81

[9] Krause, A., Horvitz, E., Kansal, A., Zhao, F.: Toward community sensing. In: IPSN '08: Proceedings of the 7th international conference on Information processing in sensor networks, Washington, DC, USA, IEEE Computer Society (2008) 481–492

[10] Ahmad, Y., Nath, S.: COLR-Tree: Communication-efficient spatio-temporal indexing for a sensor data web portal. IEEE 24th International Conference on Data Engineering (April 2008) 784–793

[11] Grosky, W., Kansal, A., Nath, S., Liu, J., Zhao, F.: SenseWeb: An infrastructure for shared sensing. IEEE Multimedia **14**(4) (Oct.-Dec. 2007) 8–13

[12] Google Traffic. `http://maps.google.com/`

[13] ZigBee Alliance. `http://www.zigbee.org/`

[14] Park, C., Chou, P.H.: Eco: Ultra-wearable and expandable wireless sensor platform. In: Third International Workshop on Body Sensor Networks (BSN'06). (April 2006)

[15] Ecomote. `http://www.ecomote.net/`

[16] Bluetooth Low Energy Technology. `http://www.bluetooth.com/Bluetooth/Products/low_energy.htm`

[17] Lee, C.H.: EcoFlock: A clustered handoff scheme for ultra-compact wireless sensor platforms in EcoPlex network. Master's thesis, National Tsing Hua University (2009)

[18] Jumbo Frame. `http://en.wikipedia.org/wiki/Jumbo_frame`

[19] Hsueh, C.H.: EcoExec: A highly interactive execution framework for ultra compact wireless sensor nodes. Master's thesis, National Tsing Hua University (2009)

[20] Cha, H., Choi, S., Jung, I., Kim, H., Shin, H., Yoo, J., Yoon, C.: RETOS: resilient, expandable, and threaded operating system for wireless sensor networks. In: IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks, New York, NY, USA, ACM (2007) 148–157